anemometer (29008 Dofs, 1-3 parameter)

****

Example with 1 parameter:

G(p) = C((sE - A_{v0}- p (A_{v1} - A_{v0}))^{-1}B)

with fluid velocity p in [0, 1]
and output as difference between the two sensors:

C(1,173) = 1,
C(1,133) = -1

permutation:
C = C*P^T

for simulation:
u is heater with step function (at id 35)

*.B load vector
*.E damping matrix
*.C permutation matrix (matrices built with ANSYS, for adapting nodes and dofs
*.A stiffness matrices

_v0 diffusion
_v1 diffusion and convection

Use mmread for reading the matrices into matlab.

****
*

Example with 3 parameters:
(infos about the matrices were extracted from IMTEK Mathematica code)

G(p_0,p_1,p_2) = C((s (Mc + p_0 Mc_var) - (Md + p_2 Md_var + p_0 p_1 Mconv))^{-1}B)

with parameters:

p_0 in [0, 1]   (c - specific heat)
p_1 in [0.1, 2] (kappa - thermal conductivity)
p_2 in [1, 2]   (c v - specific heat fluid velocity)

rho (mass density) seems to be fixed (rho = 1?)

and matrices:

```
damping
Mc      = model_0_1_1.E
Mc_var  = model_0_2_1.E - Mc

stiffness
Md      = model_0_1_1.A
Md_var  =  chop(model_1_1_1.A - Md)
+ chop(model_1_2_2.A -
model_1_2_1.A)
Mconv   = model_0_2_1.A - Md


B       = model_0_1_1.B
```

Infos from IMTEK:
Take care of too small numbers
(otherwise you will obtain
illconditioned matrices)
Thus, filter them out:

Mathematica:
Chop[expr, delta] replaces numbers
smaller in absolute magnitude than
delta by 0.
Chop uses a default tolerance of
10^-10

matlab (chop =):

```
A = A .*  (abs(A)>= 10^-12);
```

_____--
Ulrike Baur, Aug 31 2011